Hochschule Konstanz
Fakultät Informatik

Machine Learning:: Linear Methods

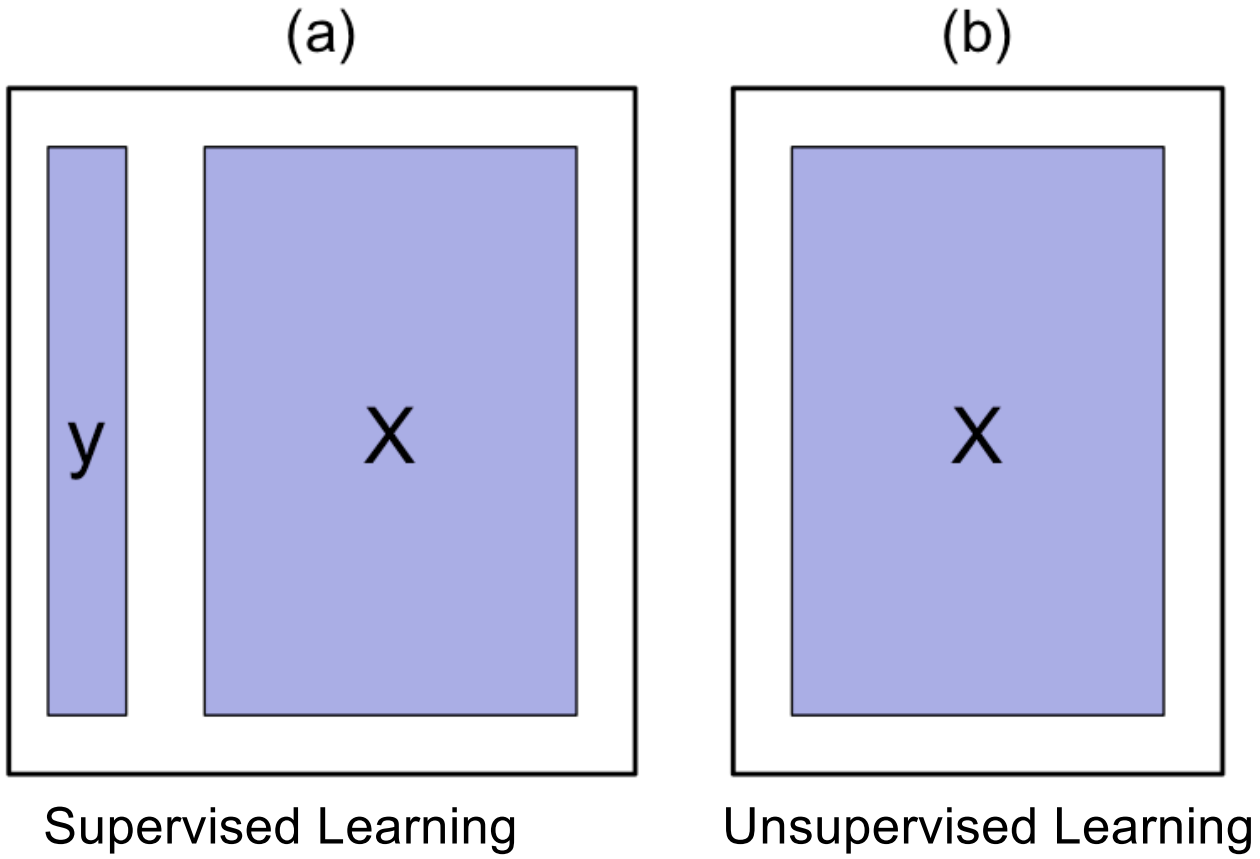
Dozenten: Prof. Dr. M. O. Franz, Prof. Dr. O. Dürr

Preliminary Slides
Work in Progress

Remark: Much of the material have been developed together with Beate Sick (ZHAW/Uni Zurich).

Supervised vs. Unsupervised Learning

- Supervised Learning: both X and Y are known
- Unsupervised Learning: only X



Supervised: Regression vs Classification

- Supervised learning problems can be further divided into regression and classification problems.

- Regression covers situations where Y is continuous/numerical. e.g.
  - Predicting the value of the Dow in 6 months.
  - Predicting the value of a given house based on various inputs.

- Classification covers situations where Y is categorical e.g.
  - Will the Dow be up (U) or down (D) in 6 months?
  - Is this email a SPAM or not?

- Some methods work well on both types of problem
  - Linear Methods
    - Linear Regression is a regression method
    - Logistic Regression (is a classification method with a strange name)
    - Perceptron (is a classification model)
  - e.g. Neural Networks

Supervised: Parametric Models

Almost all supervised learning is done with parametric Models

- Models which have parameters (a.k.a. weights)
- We have (training $i = 1 \ldots N\_\text{training}$) data in pairs $x^{(i)}$ and $y^{(i)}$

$x^{(i)}$ → model parametrized with weights → $\hat{y}^{(i)}$

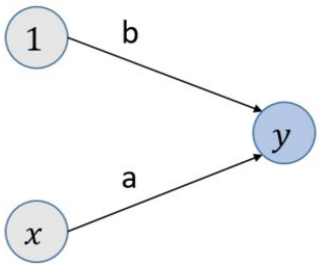
Depending on the weight the model produces a different $\hat{y}^{(i)}$

- Examples
  - Facerec.: Faces and Names
  - Age Prediction: Faces and Age (numerical problem)

- After the training, we can discard the training data (it's all in the parameters)
- How to tune the nobs that the output of the model $\hat{y}^{(i)}$ matches the "true" value $y^{(i)}$? We optimize a loss.

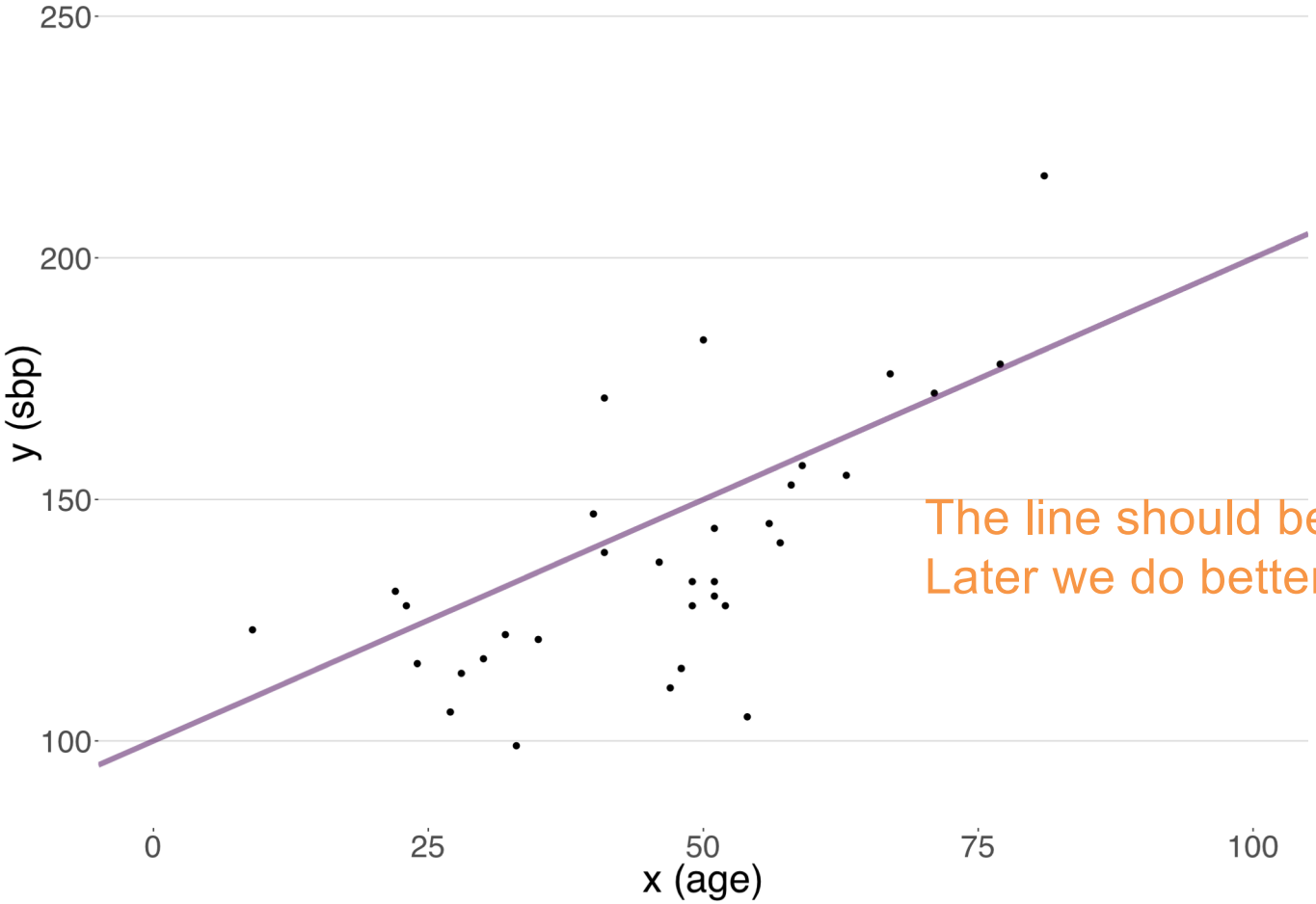
1-D Linear Regression (as curve fitting)

Example: blood pressure given age



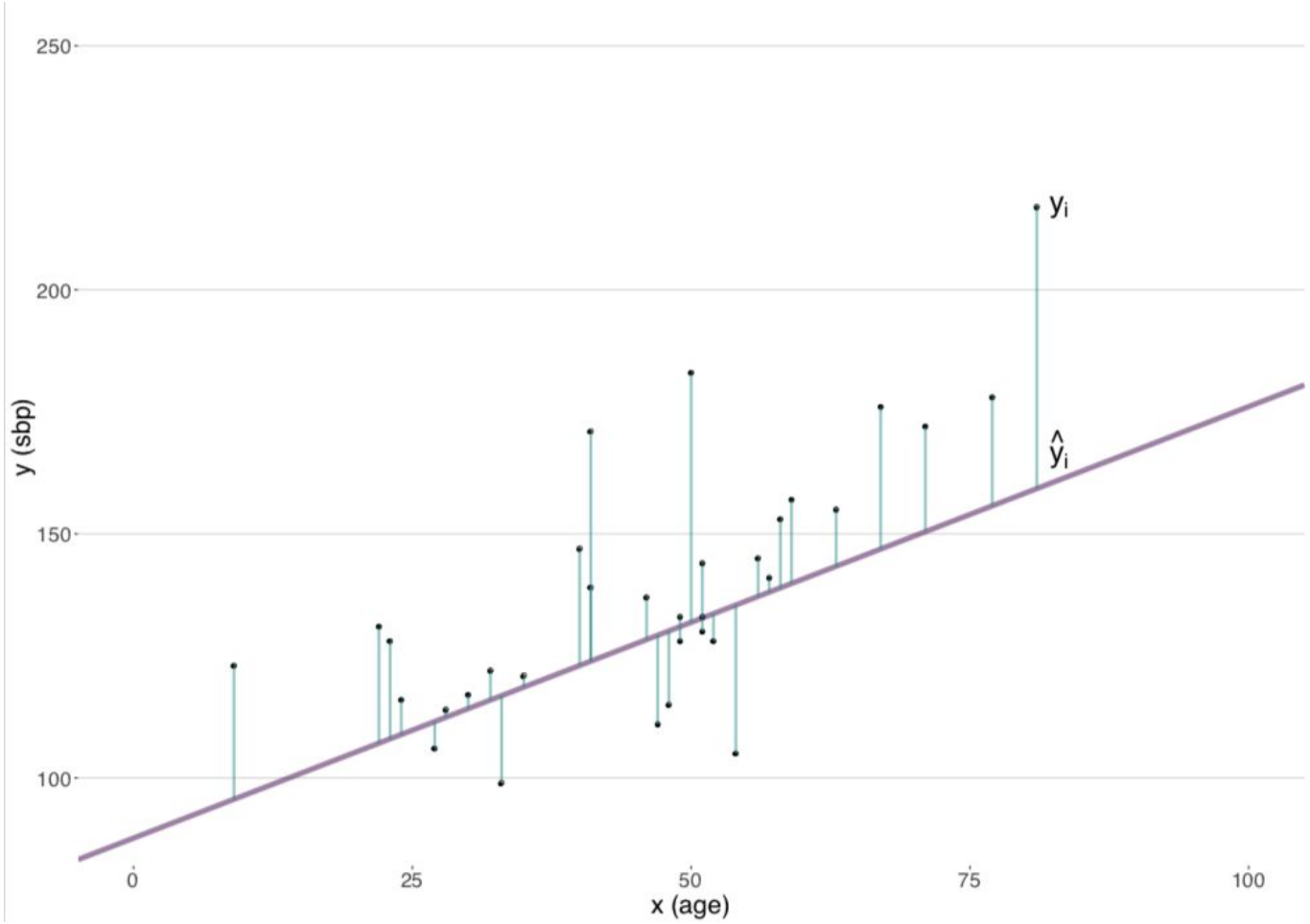
Like a neural net

Model: $\hat{y} = a \cdot x + b$



The line should be “a good guess for” y.
Later we do better.

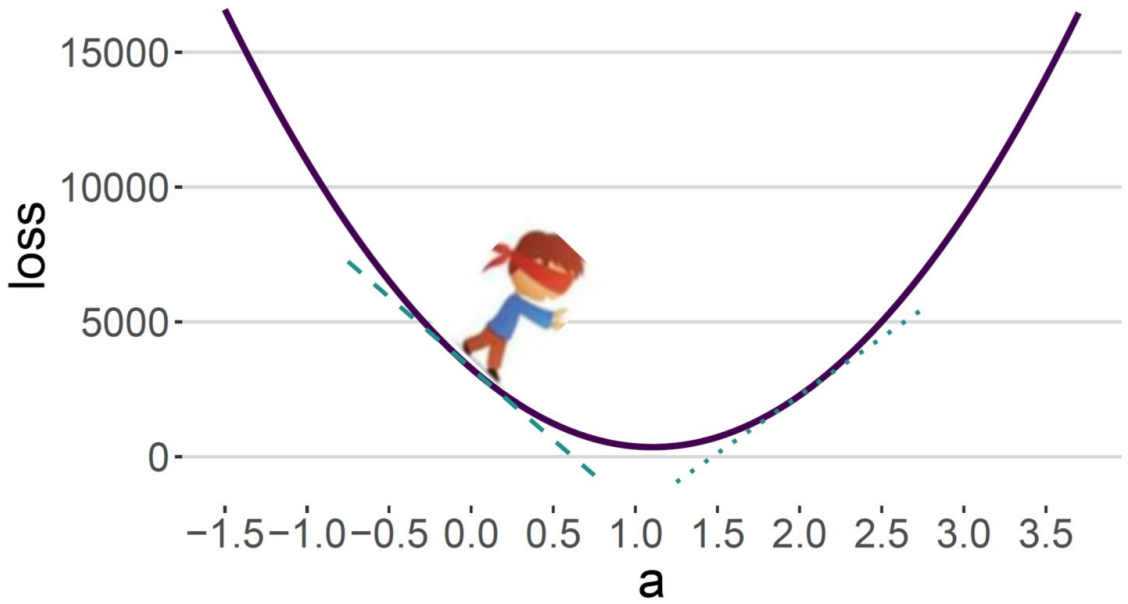
Definition of the loss



$$Loss = MSE = 1/n \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = 1/n \sum_{i=1}^{n} (y_i - (a \cdot x_i + b))^2$$

Optimizing the loss: Gradient Descent

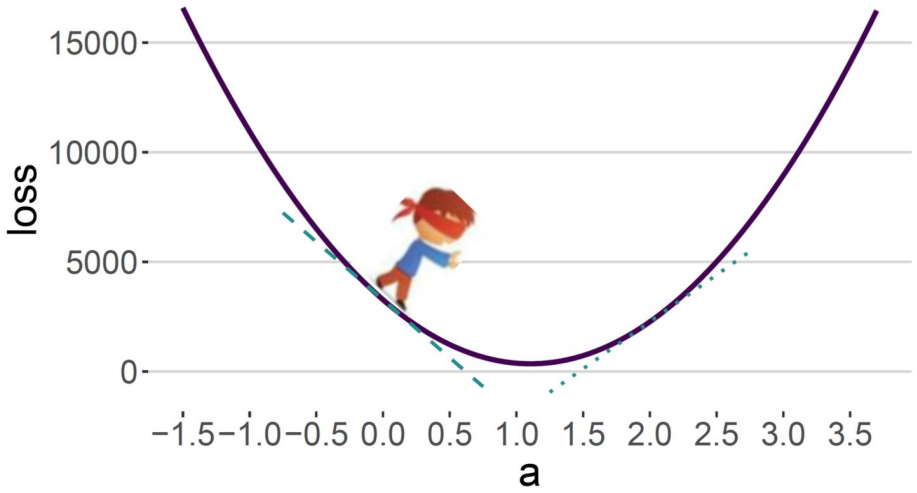
- Note for LR there is a closed from solution. We use gradient descent since it's a very general methods and works for many ML and DL algorithms.
- Fix b and only vary a



- How would you choose the size of the step [you are blind!]

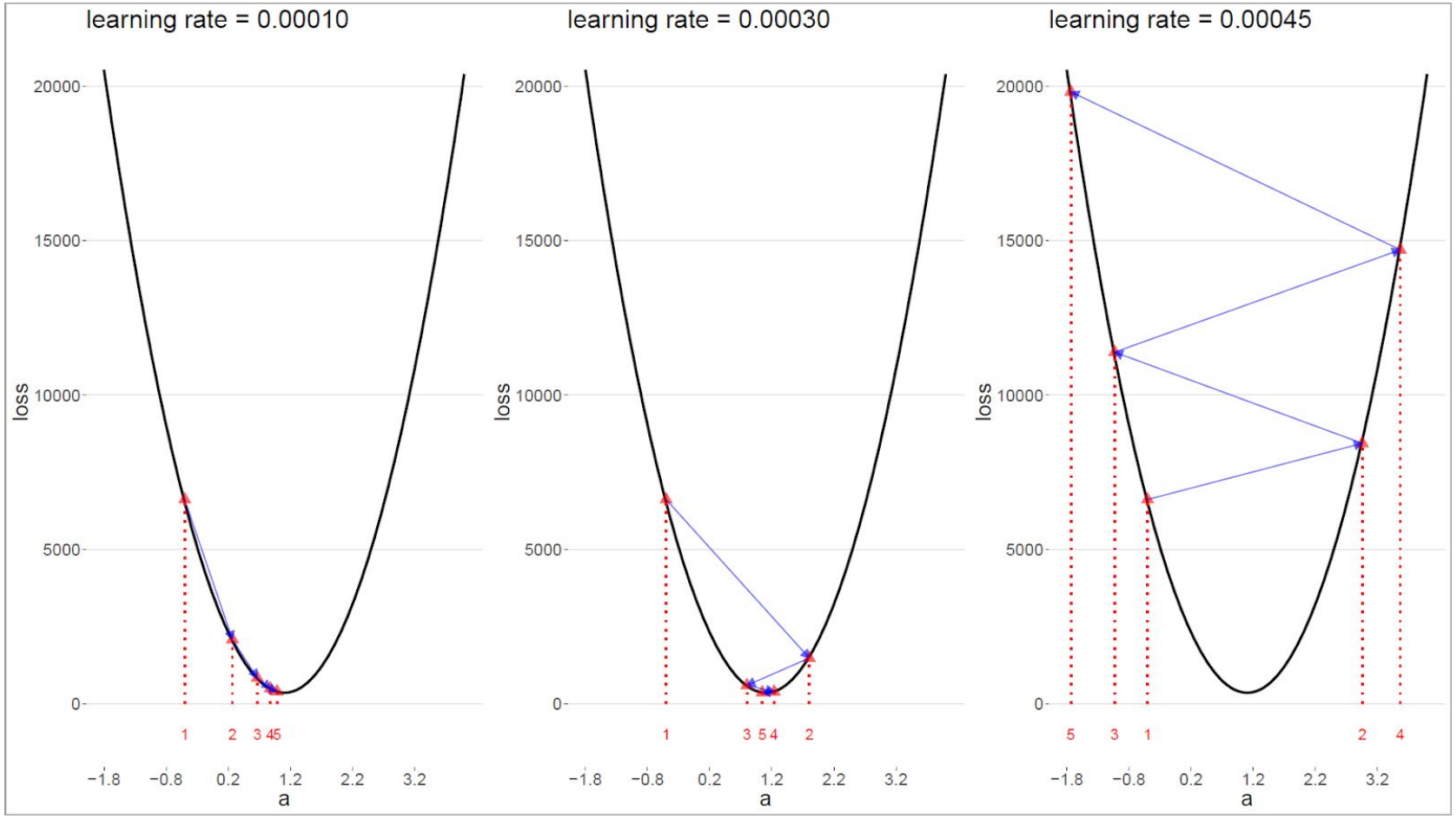
Gradient update formula

$$a_{t+1} = a_t - \eta \cdot grad_a(Loss)$$



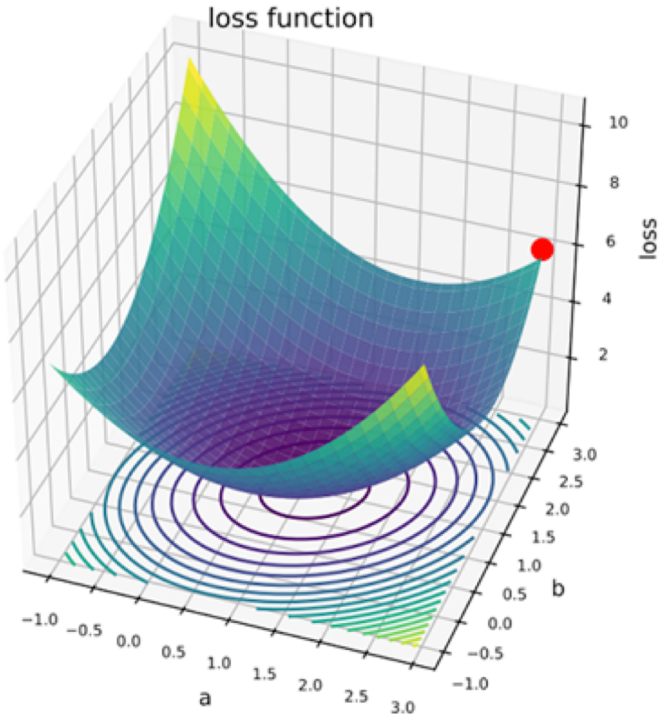
How to choose the parameter η (Learning Rate)?

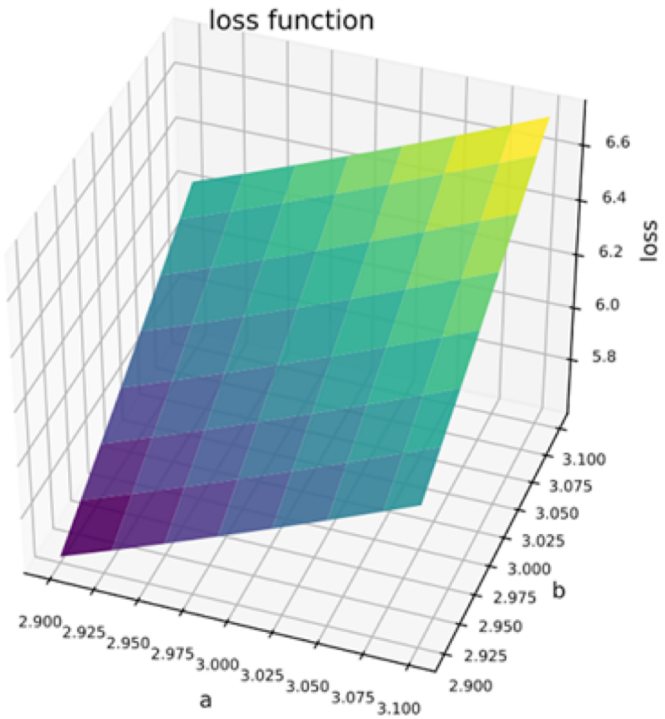
Proper learning rate

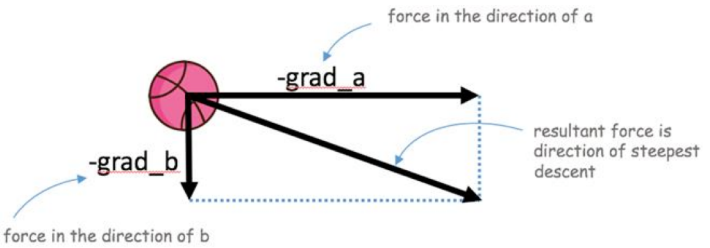


See: https://developers.google.com/machine-learning/crash-course/fitter/graph

In two dimensions







$$a_{t+1} = a_t - \eta \cdot grad_a$$

$$b_{t+1} = b_t - \eta \cdot grad_b$$

In vector notation: $W^{t+1} = W^T - \eta \cdot \nabla W$

Dangers of Gradient Descent in 2D

Many Problems in ML are convex. DL magically also works for non-convex problems

Multivariate Linear Regression (as curve fitting)

See notebook linear_methods/LinearRegression

Example

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	9.7
197	177.0	9.3	6.4	12.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	13.4

200 rows × 4 columns

$$\widehat{sales} = 1 \cdot w_o + w_1 TV + w_2 Radio + w_3 Newspaper$$

Multivariate Linear regression

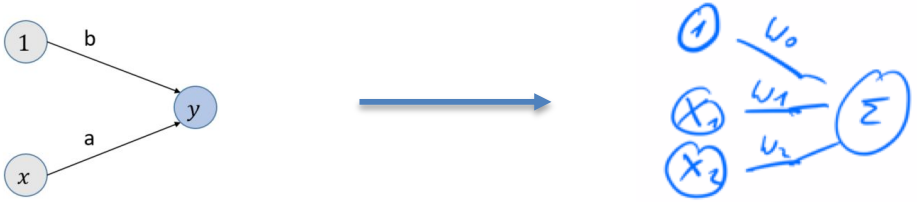
- Simple model
  - $\hat{y} = f_\theta(x) = 1 \cdot w_o + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d = \vec{w}^t \vec{x} + w_0 =< w, x > + w_0$

- A note on notation
  - w_0 is often denoted with *b* the bias (extended data matrix)
  - Often the bias is included in $\vec{w}^t = (w_o, w_1, w_2, \cdots, w_d)$
    - $\hat{y} = \vec{w} \cdot \vec{x}$
  - In some literature (Tibshirani) $\vec{w} = \vec{\beta}$
  - Often no vector symbols

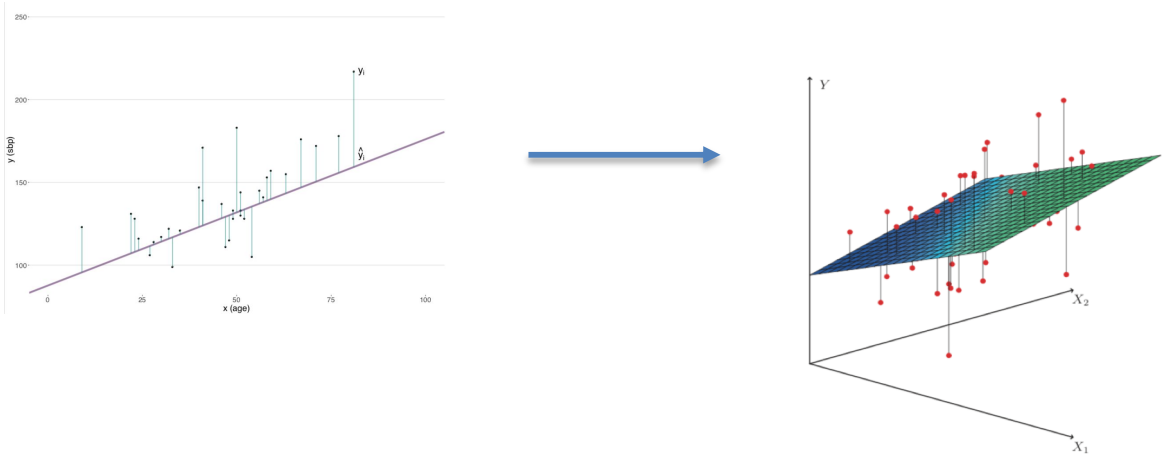
Multivariate Linear regression

- Simple model
  - $\hat{y} = f_\theta(x) = 1 \cdot w_o + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d = \boldsymbol{w^T x} + w_0 =< \boldsymbol{w}, \boldsymbol{x} > + w_0$

- As a NN



- Geometric interpretation $f_\theta(x)$ is a plane



Matrix notation

Design Matrix

Intercept	TV	Radio	Newspaper
1.0	230.1	37.8	69.2
1.0	44.5	39.3	45.1
1.0	17.2	45.9	69.3
1.0	151.5	41.3	58.5
1.0	180.8	10.8	58.4
...	...	...	...
1.0	38.2	3.7	13.8
1.0	94.2	4.9	8.1
1.0	177.0	9.3	6.4
1.0	283.6	42.0	66.2
1.0	232.1	8.6	8.7

$$\begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix} =$$

y_hat
20.244783
12.426815
12.310875
17.457258
13.204685
...
5.815722
8.534241
13.000720
23.386213
15.297458

Sales
22.1
10.4
9.3
18.5
12.9
...
7.6
9.7
12.8
25.5
13.4

$$X \cdot w = \hat{y}$$

loss = tf.reduce_mean((y - y_hat)**2) #True sales is y

Solving the regression problem

Notation mit Daten matrix — Erweiterte Datenmatrix die Bias beinhaltet

$$L = \sum_i \left(\sum_j x_{ij} \cdot w_j - y_i \right)^2 \qquad \nabla_i$$

$$\frac{\partial L}{\partial w_\ell} = \sum_i 2 \left(\sum_j x_{ij} w_j - y_i \right) \sum_j$$

$$= \sum_i 2 \left(\sum_j x_{ij} w_j - y_i \right) x_{i\ell}$$

$$= \sum_i 2 \left(\underbrace{\sum_j x_{ij} w_j}_{(X \cdot w)_i} - y_i \right) x_{i\ell}$$

$$= \sum_i 2 (X \cdot w - y)_i \, x_{i\ell} = 0$$

$$= \sum_i 2 (X^T)_{\ell i} (X \cdot w - y)_i$$

$$= 2 \left(X^T \cdot (X \cdot w - y) \right)_\ell = 0$$

$$\nabla L = 2 X^T (X \cdot w - y)$$

↑ Diese Formel kann man auch nochmal für den Gradient Descent nehmen

$$\nabla L \stackrel{!}{=} 0$$

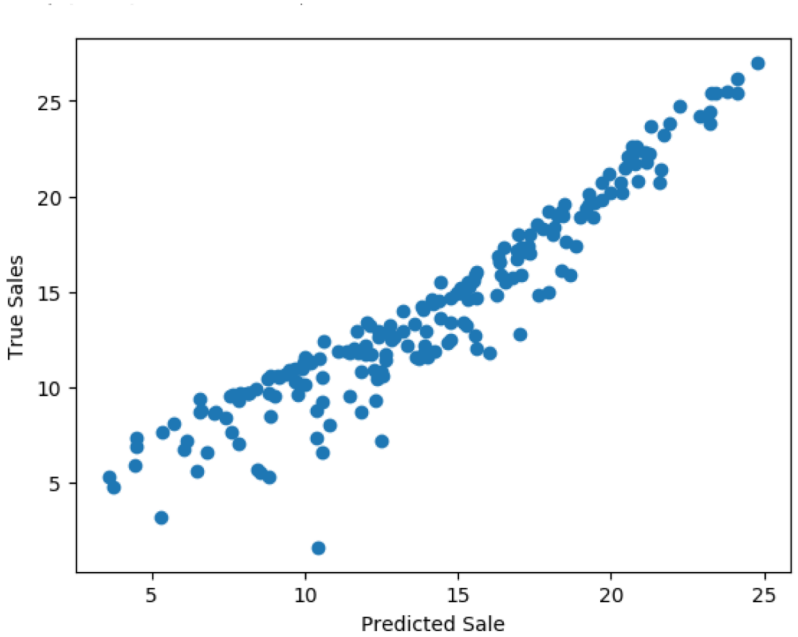
$$\Rightarrow X^T X w = X^T y$$

$$\Rightarrow w = (X^T X)^{-1} X^T \cdot y$$

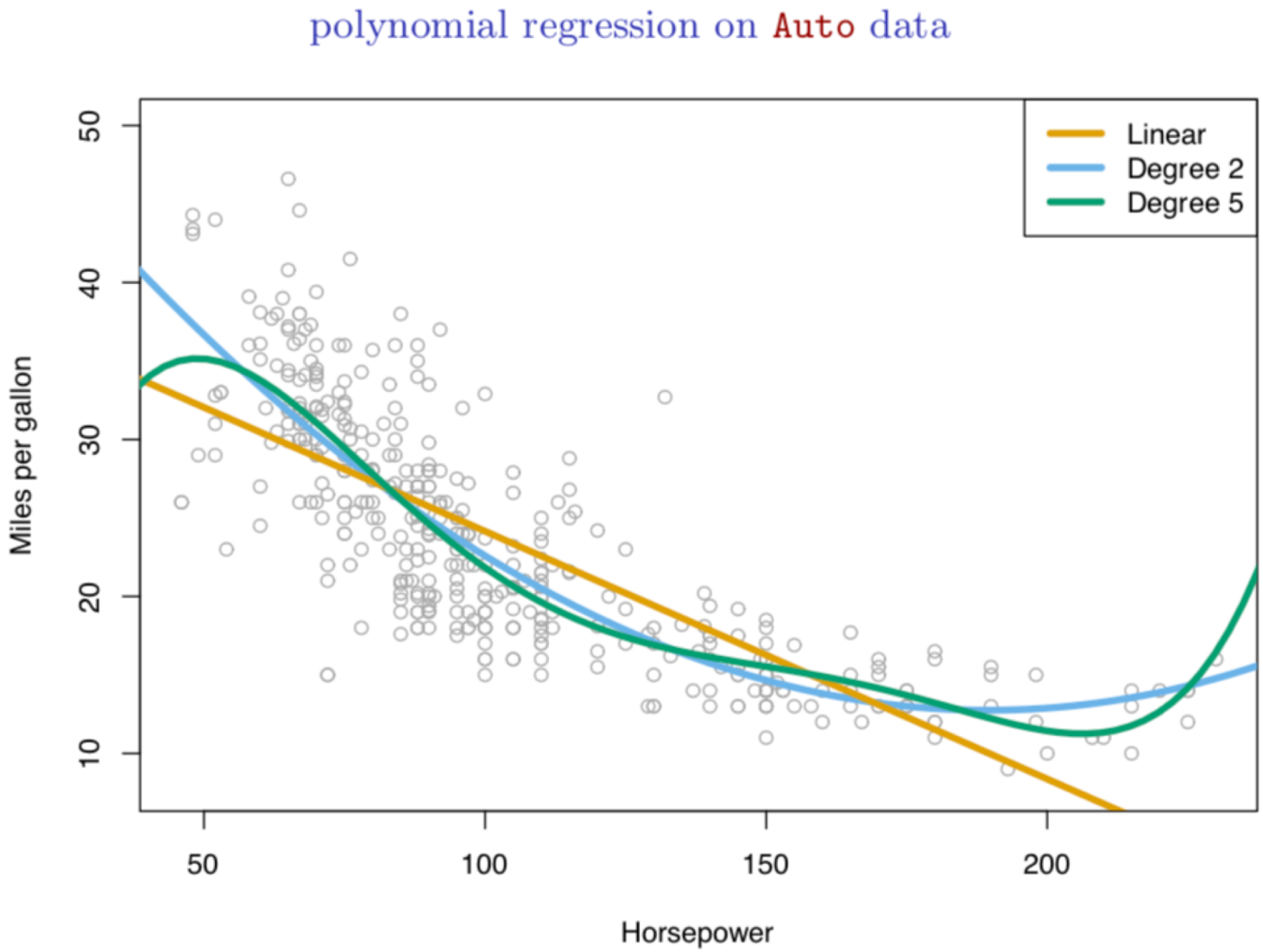
Final Formula $\hat{\beta} = W = (X^T X)^{-1} X^T y$

Demo

- Stepping through the notebook



Non Linear Effect / Polynomial Data



Non Linear Effect / Polynomial Data

The figure suggests that

$$\texttt{mpg} = \beta_0 + \beta_1 \times \texttt{horsepower} + \beta_2 \times \texttt{horsepower}^2 + \epsilon$$

may provide a better fit.

	Coefficient	Std. Error	t-statistic	p-value
Intercept	56.9001	1.8004	31.6	< 0.0001
horsepower	-0.4662	0.0311	-15.0	< 0.0001
horsepower^2	0.0012	0.0001	10.1	< 0.0001

Duale Formulierung der linearen Regression

Die Lösung des Regressionsproblems läßt sich ähnlich wie beim Perzeptron als Linearkombination der Trainingsbeispiele schreiben:

$$\hat{\mathbf{w}} = (X^\top X)^{-1} X^\top \mathbf{y} = (X^\top X)^{-1} \sum_{i=1}^{\ell} y_i \hat{\mathbf{x}}_i = \sum_{i,j=1}^{\ell} (X^\top X)^{-1}_{ij} y_i \hat{\mathbf{x}}_i,$$

d.h. $\hat{\mathbf{w}} = X^\top \boldsymbol{\alpha}$ mit Einbettungstärken $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots)^\top$. Einsetzen in die quadratische Fehlerfunktion

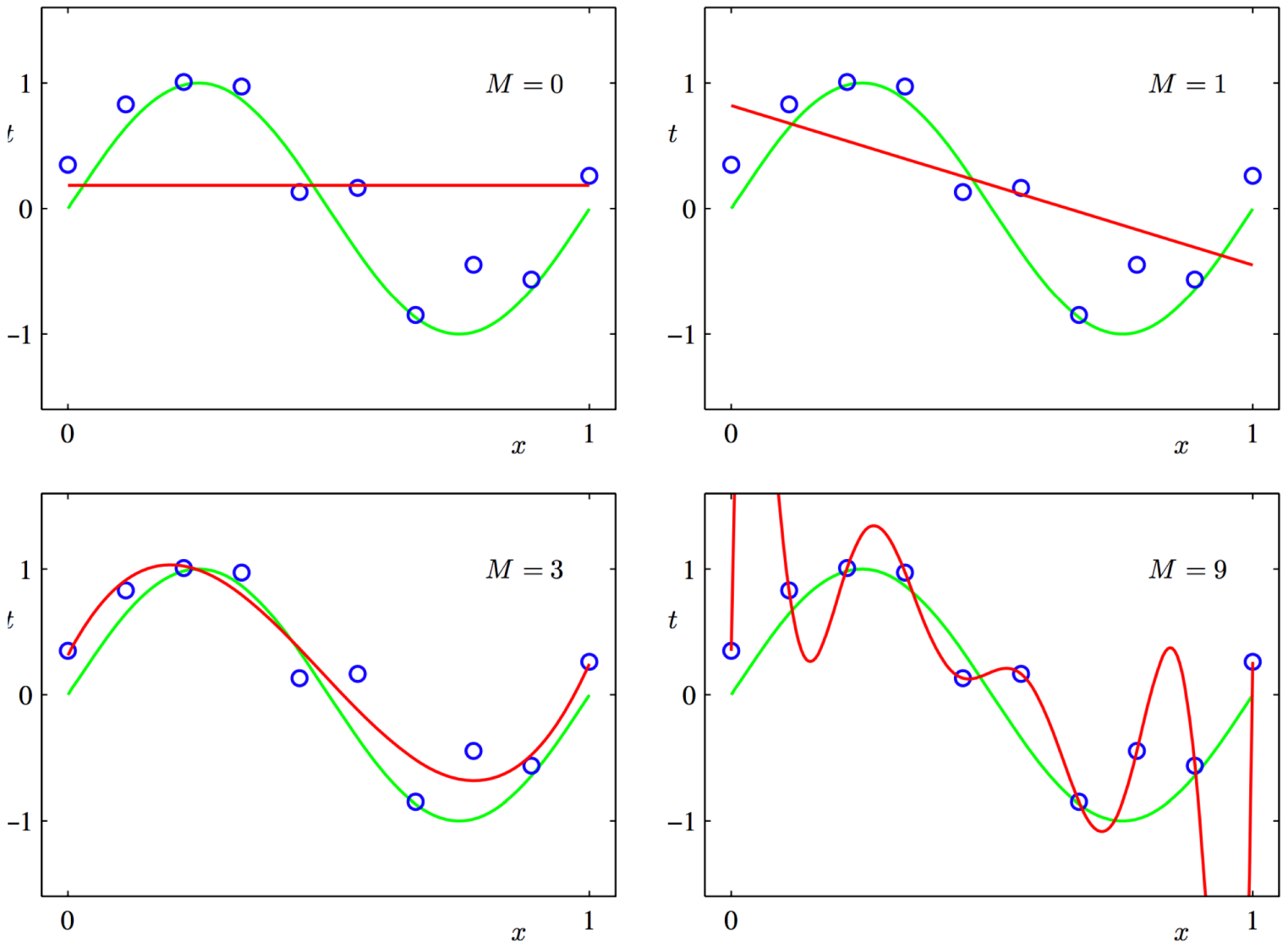
$$\begin{aligned} L(\hat{\mathbf{w}}) &= \mathbf{y}^\top \mathbf{y} - 2\mathbf{y}^\top X \hat{\mathbf{w}} + \hat{\mathbf{w}}^\top X^\top X \hat{\mathbf{w}} \\ &= \mathbf{y}^\top \mathbf{y} - 2\mathbf{y}^\top X X^\top \boldsymbol{\alpha} + \boldsymbol{\alpha}^\top X X^\top X X^\top \boldsymbol{\alpha} \\ &= \mathbf{y}^\top \mathbf{y} - 2\mathbf{y}^\top G \boldsymbol{\alpha} + \boldsymbol{\alpha}^\top G G \boldsymbol{\alpha} \end{aligned}$$

mit Gram-Matrix $G = G^\top = X X^\top = \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$. Ableiten nach $\boldsymbol{\alpha}$ und Nullsetzen ergibt die Lösung

$$\boldsymbol{\alpha} = (GG)^{-1} G \mathbf{y} = G^{-1} \mathbf{y}.$$

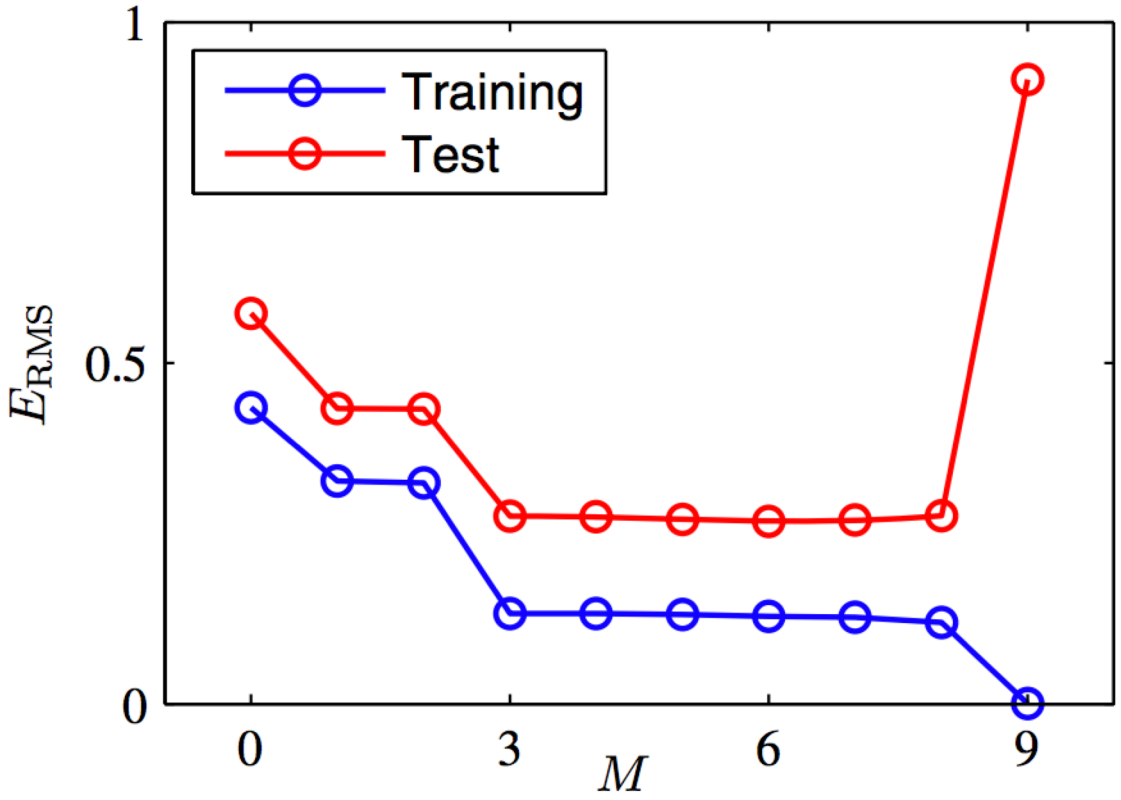
Ridge Regression

Overfitting



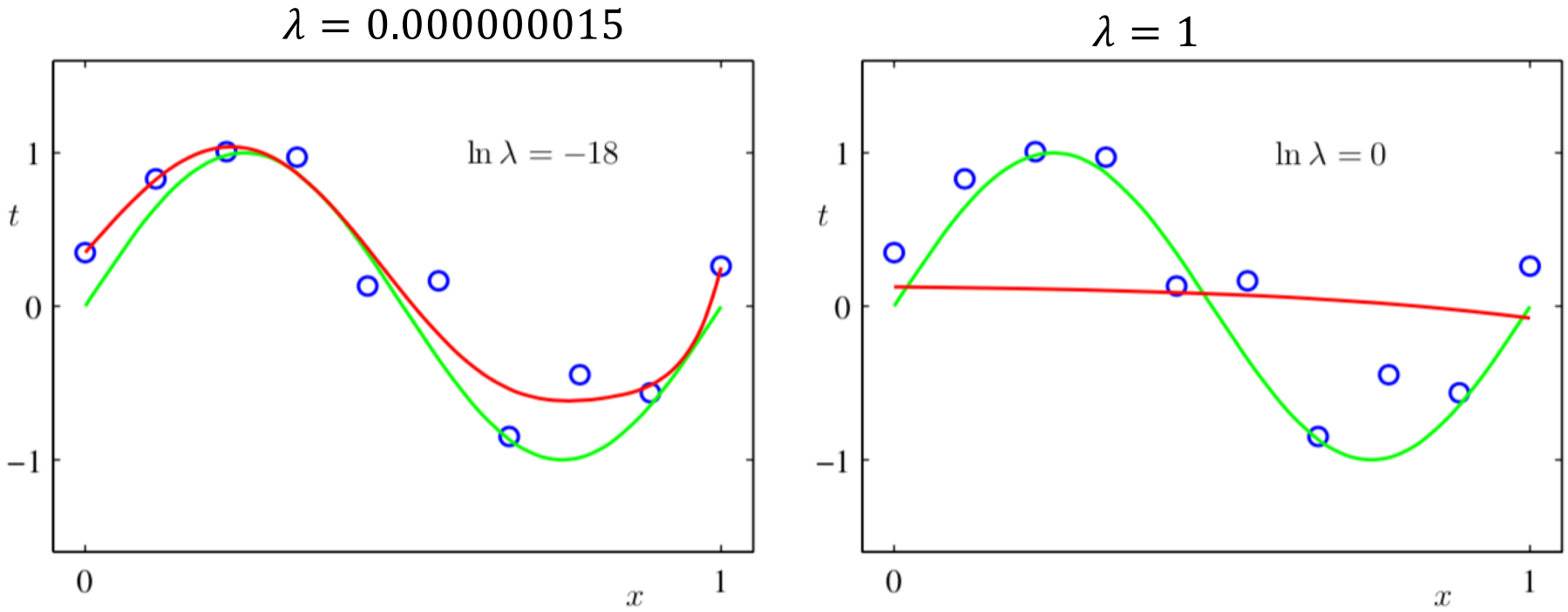
Expand the data matrix to columns $x^0, x^1 \ldots x^M$

Overfitting



More on overfitting in later lectures …

Ridge Regression



$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathrm{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2,$$

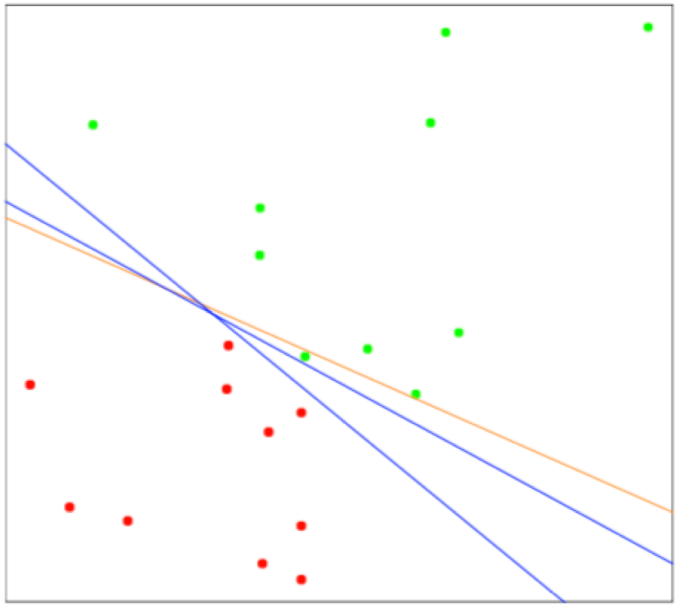
Additional loss term L2-Regularisation

- It costs to have non-zero parameters.
- Parameters are closer to zero (*shrinkage*)
- Ridge Regression reduces overfitting.
- Allows to fit linear regression to #parameters > # Data problems
- Also popular Lasso L1 regularisation

Perceptron

Perceptron

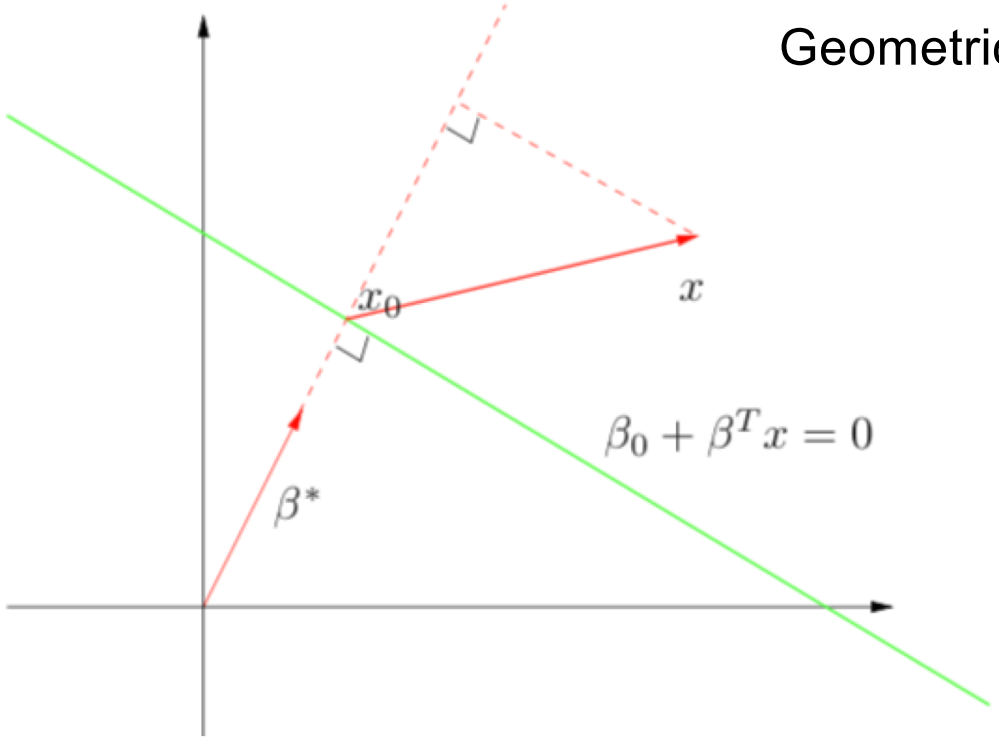
- The simplest way for a linear machine to do classification
  - Other approach is logistic regression
- We have binary classification problem
  - Code $y_i = \pm 1$
- Based on a geometric concept of hyperplanes
  - Needs classes which are separable by hyperplanes
  - Will be extended in the Support Vector Machine



Orange line is linear regression

Figure: ESLR chapter 4.5

Geometric properties



1. For any two points x_1 and x_2 lying in L, $\beta^T(x_1 - x_2) = 0$, and hence $\beta^* = \beta/||\beta||$ is the vector normal to the surface of L.

2. For any point x_0 in L, $\beta^T x_0 = -\beta_0$.

3. The signed distance of any point x to L is given by

$$\begin{aligned} \beta^{*T}(x - x_0) &= \frac{1}{\|\beta\|}(\beta^T x + \beta_0) \\ &= \frac{1}{||f'(x)||} f(x). \end{aligned} \tag{4.40}$$

Figure: ESLR chapter 4.5

Definition of a loss function

- $\frac{x_i^T \beta + \beta_0}{||\beta||}$ is the signed distance to the hyperplane
- For examples i with $y_i = 1$ this should be positive
- For examples i with $y_i = -1$ this should be negative

Loss function

$$\text{loss} = \sum_i \max(0, y_i(x_i^T \beta + \beta_0))$$

- Correct point gets zero, incorrect gets -1
- Loss = zero for separating plane
- > 0 for non-separating plane
- Taking Gradient:

$$\partial \frac{D(\beta, \beta_0)}{\partial \beta} = -\sum_{i \in \mathcal{M}} y_i x_i,$$

$$\partial \frac{D(\beta, \beta_0)}{\partial \beta_0} = -\sum_{i \in \mathcal{M}} y_i.$$

Learning Rule [in python]

Calculating the gradient by hand

Only visit wrongly classified points. Take gradient wrt w and b extra.

$$\frac{\partial l}{w} = -y \cdot w$$

$$\frac{\partial l}{b} = -y$$

The gradient descent update formula becomes:

```
:  def update_params(X,y,w,b):
     didUpdate = False
     for ys,xs in zip(y,X): #some python show off
       if (ys * (np.dot(xs, w) + b) < 0):
         w = w + ys*xs#eta = 1
         b = b + ys*1
         didUpdate = True
     return w,b, didUpdate
```

Algorithm converges if problem is seperable:

See https://github.com/iosKn/mldl_htwg/tree/master/perceptron

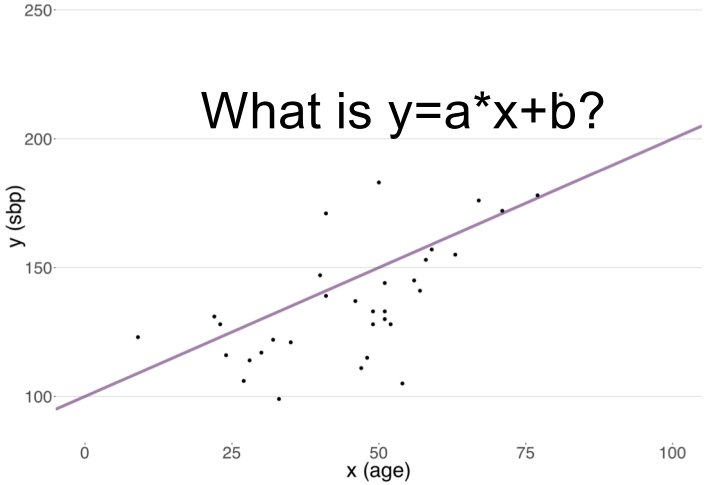
Probabilistic Treatment

Linear Models so far

- We introduced
  - linear regression to predict a value $\hat{y} \in \mathbb{R}$ form data x
  - Perceptron for binary classification $\hat{y} = \pm 1$

x → model parametrized with weights w → $\hat{y}$

- Example Linear Regression



- Perceptron
  - How sure is the classifier
  - Which line to use

Why probabilistic treatment?

- Inferences from data are intrinsically uncertain.
- Probability theory: model uncertainty instead of ignoring it.
- Unifying framework
  - Maximum Likelihood principle for loss function
  - Many ML Algorithms can be see as probabilistic models
  - Most of DL can be seen in this framework
- Allows for generalizations
    - Linear Regression → Logistic Reg., Multinomial Regression, Count Data

Probabilistic Modelling

- We model a random variable $Y_{|x}$

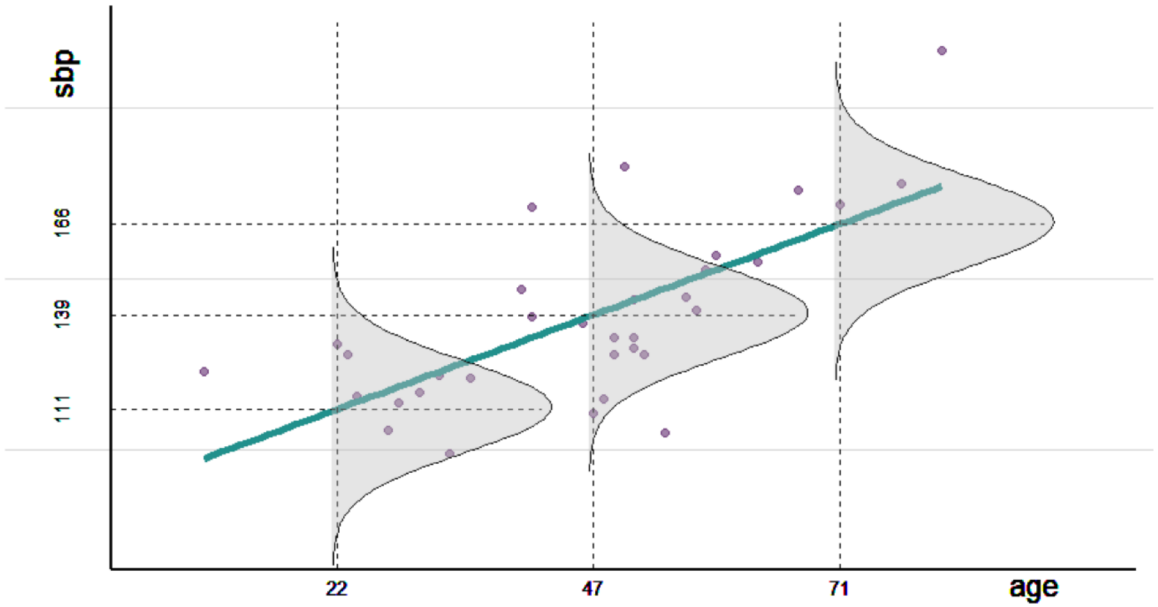
x →model parametrized with weights w→parameters of distribution

- Example linear regression
  - $p_{model}(y|x,w) = Norm(\mu = w \cdot x, \sigma = const.)$

Lineare Regression as probabilistic model

Probability Distribution for Data y given x.

For Standard Linear Regression:
Gaussian with $\mu = a * x + b$ and $\sigma = constant$



$$Y_{x_i} \sim N(\mu_{x_i}, \sigma^2)$$

Don't be confused σ
is often treated as constant

How to determine the parameters of a probabalistic model

Read section MaxLike in Introduction

Maximum Likelihood
(one of the most beautiful ideas in statistics)

Ronald Fisher in 1913
Also used before by
Gauss, Laplace

Likelihood / “probability“
(often known)

$$M(\theta) \longrightarrow \text{Data}$$

Tune the parameter(s) θ of the model M so that (observed) data is most likely

What‘s the likelihood of the data for lin. regression...

Derivation of the MSE

See also blackboard

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \left\{ \prod_{l=1}^{L} \Pr(y_l \,|\, \mathbf{x}_l, \theta) \right\}$$

$w^T = a$

$$= \underset{\theta}{\operatorname{argmax}} \left\{ \prod_{l=1}^{L} \operatorname{Norm}_{y_l}(\mathbf{w}^T \mathbf{x}_l + b, \sigma^2) \right\}$$

$$= \underset{\theta}{\operatorname{argmax}} \left\{ \prod_{l=1}^{L} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(\mathbf{w}^T \mathbf{x}_l + b - y_l)^2}{2\sigma^2} \right) \right\}$$

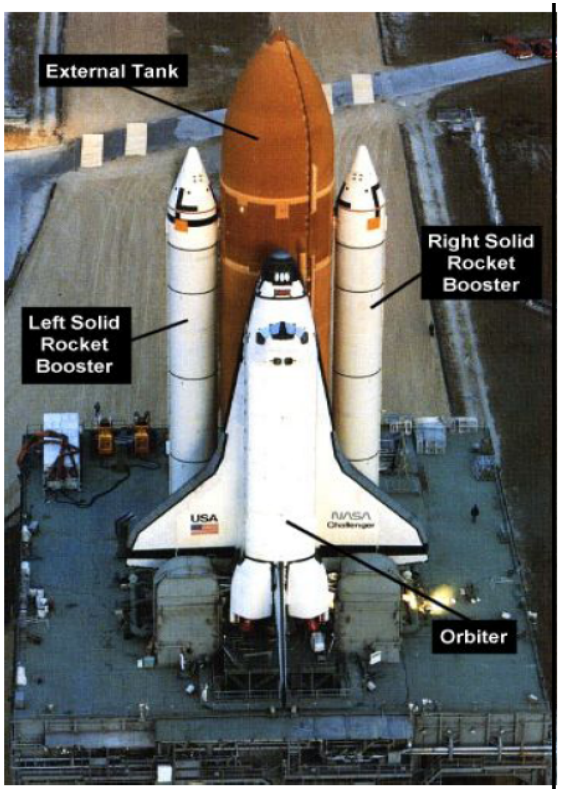
$$= \underset{\theta}{\operatorname{argmin}} \left\{ -\log \left(\prod_{l=1}^{L} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(\mathbf{w}^T \mathbf{x}_l + b - y_l)^2}{2\sigma^2} \right) \right) \right\}$$

$$= \underset{\theta}{\operatorname{argmin}} \left\{ \sum_{l=1}^{L} -\log \left(\frac{1}{2\pi\sigma^2} \right) + \frac{1}{2\sigma^2} (\mathbf{w}^T \mathbf{x}_l + b - y_l)^2 \right\}$$

$$= \underset{\theta}{\operatorname{argmin}} \left\{ \frac{1}{2\sigma^2} \sum_{l=1}^{L} (\mathbf{w}^T \mathbf{x}_l + b - y_l)^2 \right\}$$

Logistic Regression

Statistik & Challenger Desaster [motivation of logistic]



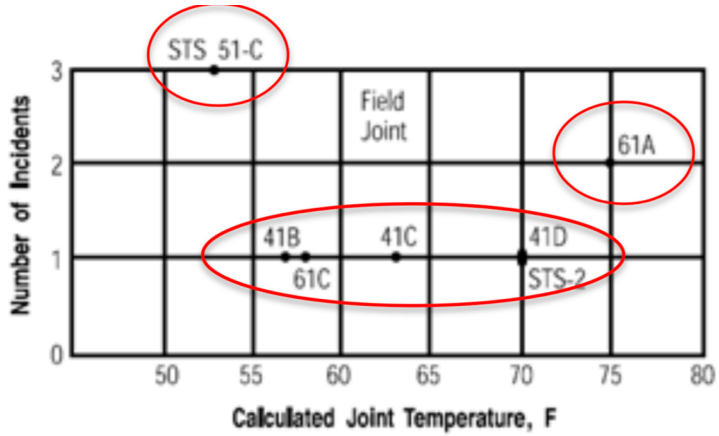
Die bemannte Raumfähre Challenger explodierte 1986 nach dem Start, weil die Dichtungsringe an den Boostern nicht dicht hielten.

Untersuchungskommission (Rogers Commision)
Richard Feynman (Nobel Preisträger in Physik), demonstriert die fehlende Elastizität Dichtungsringes bei kleinen Temperaturen.
Wie konnte man das übersehen …

Statistik & Challenger Desaster [side track]

- On the day of the challenger launch it was cold: 31°F.
- In 7 from 23 flights there have been problems with the booster bearings

Ambient temperature	Number of O–rings damaged	$\hat{p}$
53°	2	.333
57°	1	.167
58°	1	.167
63°	1	.167
70°	1	.167
70°	1	.167
75°	2	.333



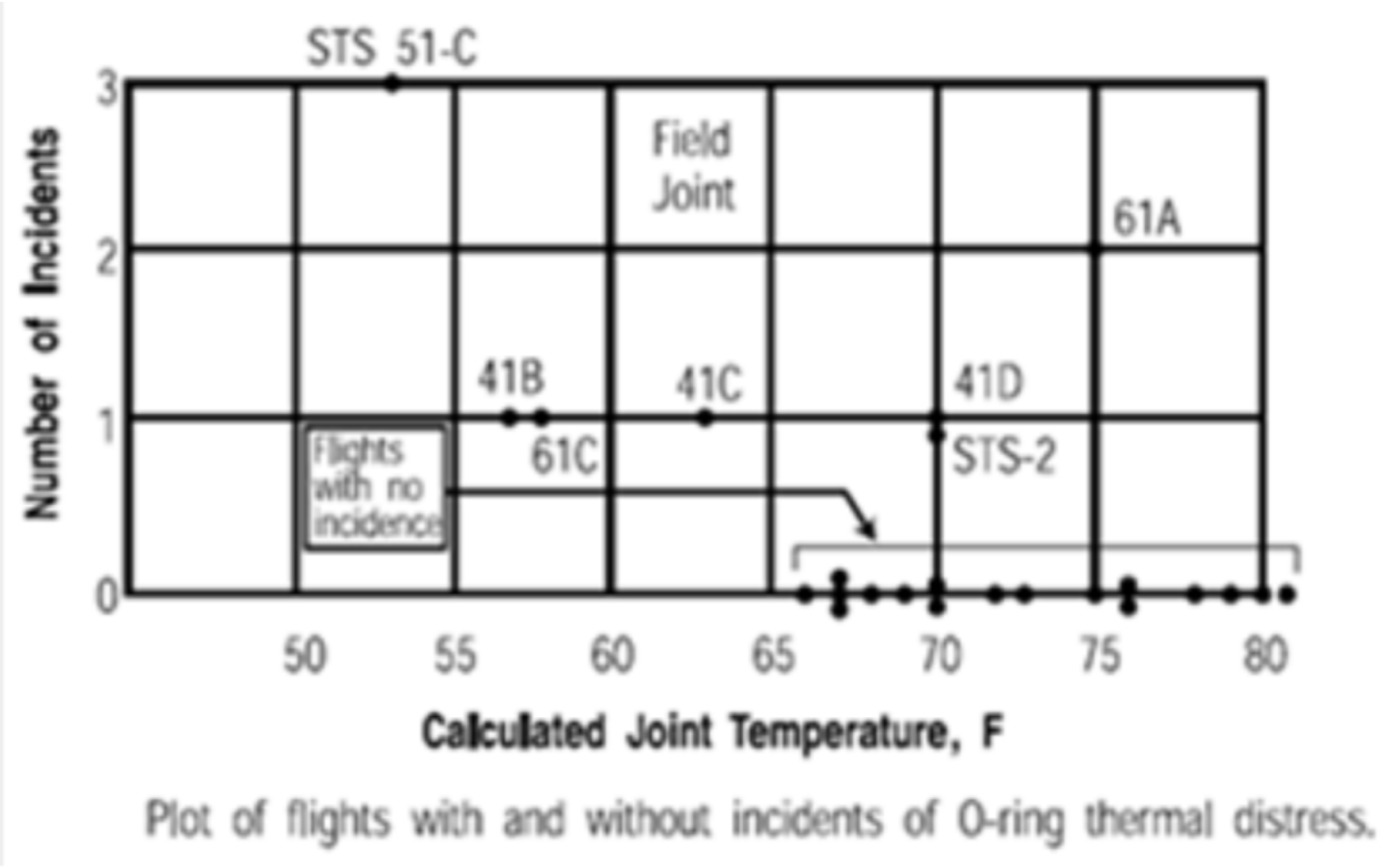
Plot of flights with incidents of O-ring thermal distress as function of temperature.

Figures from: [PRESIDENTIAL COMMISSION on the Space Shuttle Challenger Accident](https://history.nasa.gov/rogersrep/v4part3.htm) (https://history.nasa.gov/rogersrep/v4part3.htm)

- Is there an increased risk of failure at low temperatures?
  - NASA Engineer: „...I can't get a correlation between O-ring erosion, blow-by an O-ring, and temperature. “
- Would you launch (give reasons)?

Statistik & Challenger Desaster [side track]

- There is information in the successful flights

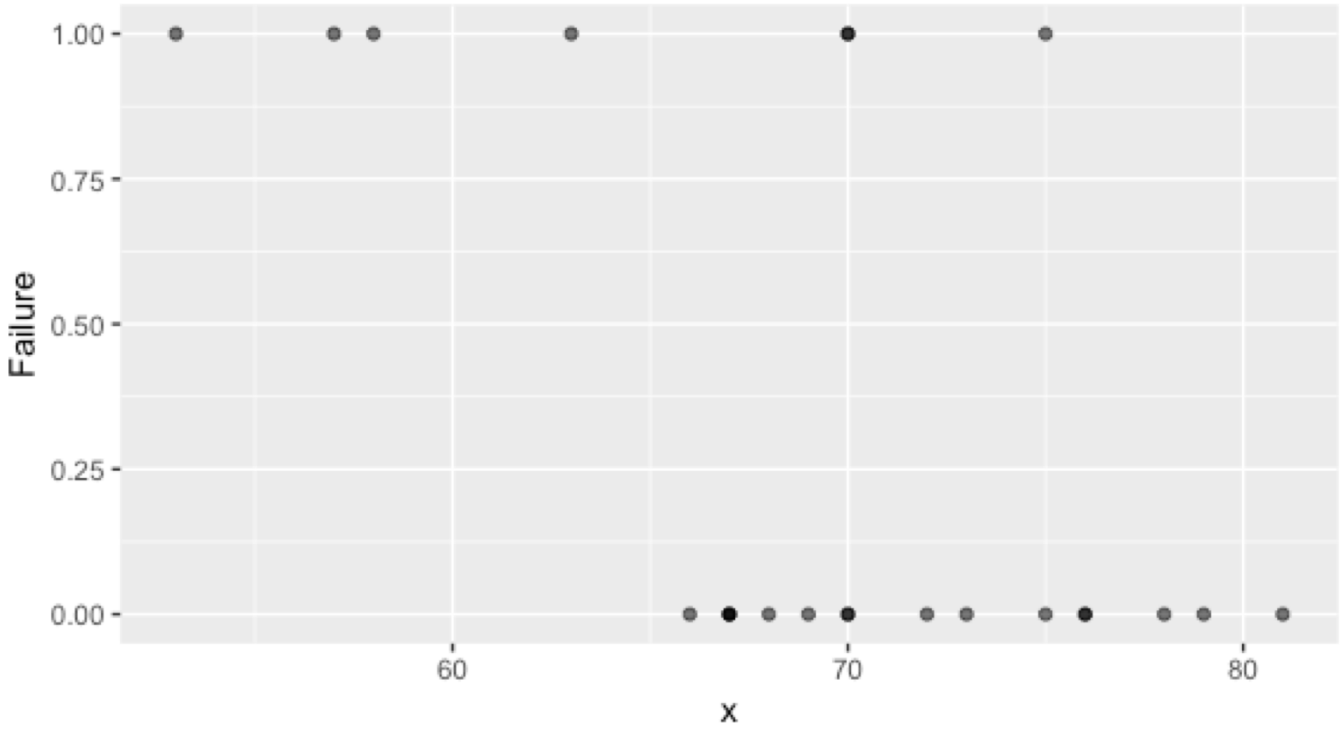


Plot of flights with and without incidents of O-ring thermal distress.

Modelling with logistic regression

Binarize to a zero / one classification

Want: $p(x) = Pr(Y = 1|x)$
Prob. for a O-ring to be defect Y=1 at a given temperature X



Question:

- **Guess curve?**
- **Why is p(X) = b +a X (linear regression) wrong?**

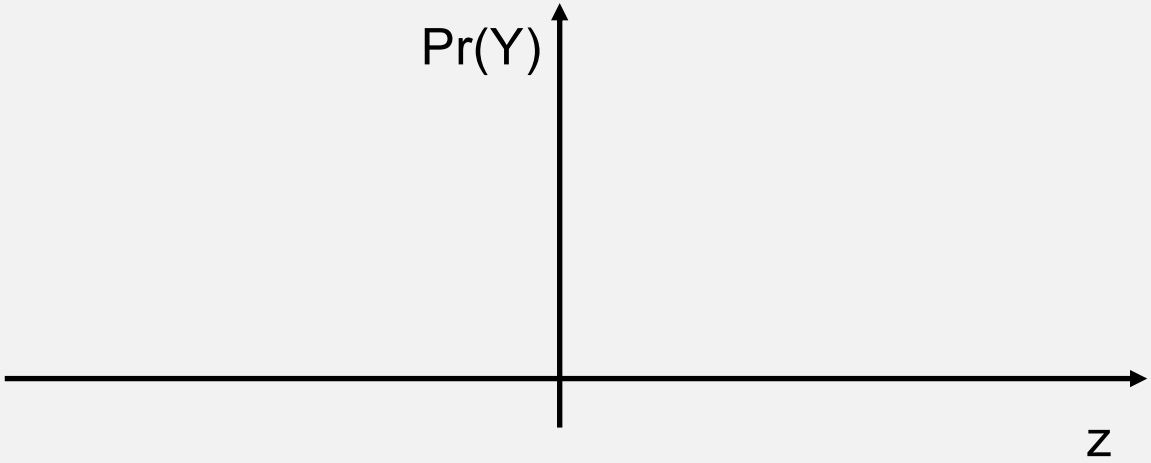
Find a suitable squeezing function

- Idea of logistic regression
- Take output of linear regression

  $-z = a \cdot x + b \quad z \in [-\infty, \infty]$

- and squeeze it to [0 and 1]

- **Task: Draw a function which could do that.**
  - Discuss with your neighbor

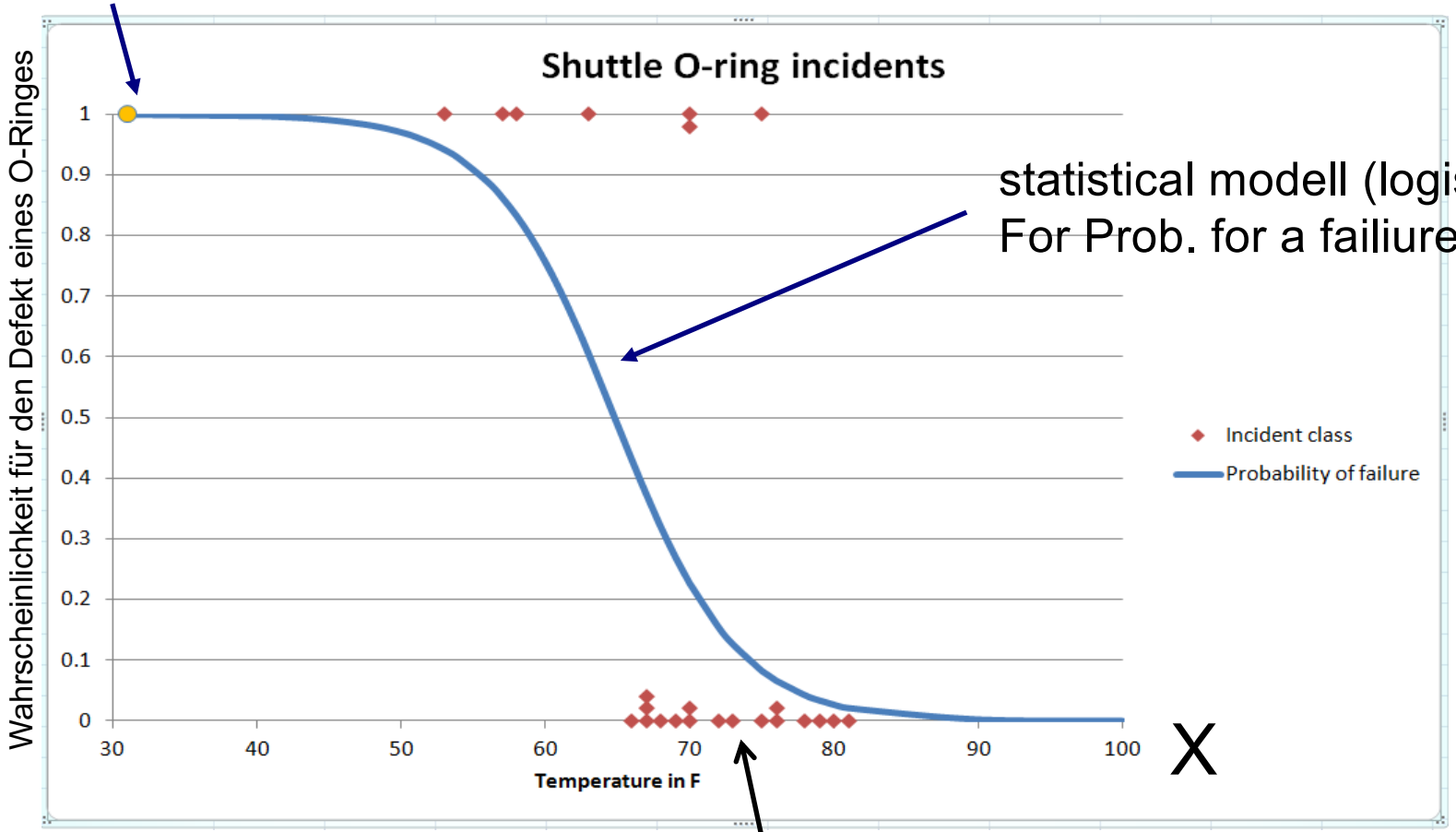


Logistic Regression: Example challenger O-rings

Predict if O-Ring is broken, depending on temperature

Challenger launch @31 F

Prob. of a failure=0.9997



P(Y=1|X)

statistical modell (logistic regression)
For Prob. for a failiure (Y=1)

X

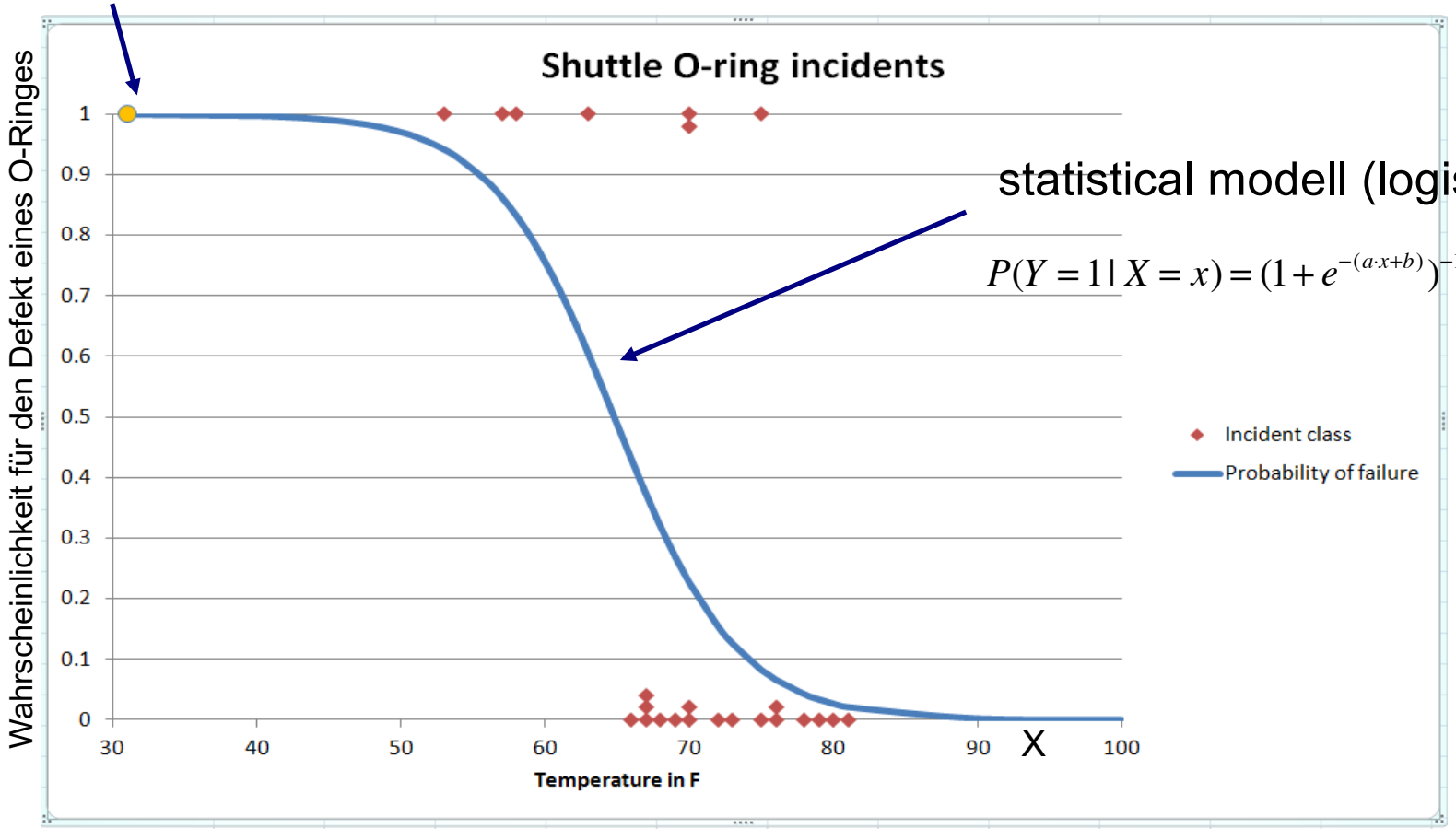
No failiure Y=0

Logistic Regression

Predict if O-Ring is broken, depending on temperature

Challenger launch @31 F

Prob. of a failure=0.9997



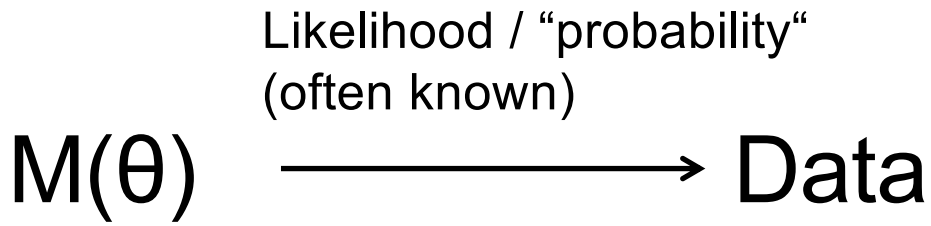
statistical modell (logistic regression)

$$P(Y=1 \mid X=x) = (1+e^{-(a \cdot x+b)})^{-1} = (1+e^{-z})^{-1} = f(x)$$

$$z \in]-\infty,\infty[\rightarrow [0,1]$$

How do we determine the parameters (a,b) of the model? M(β)

Maximum Likelihood
(one of the most beautiful ideas in statistics)

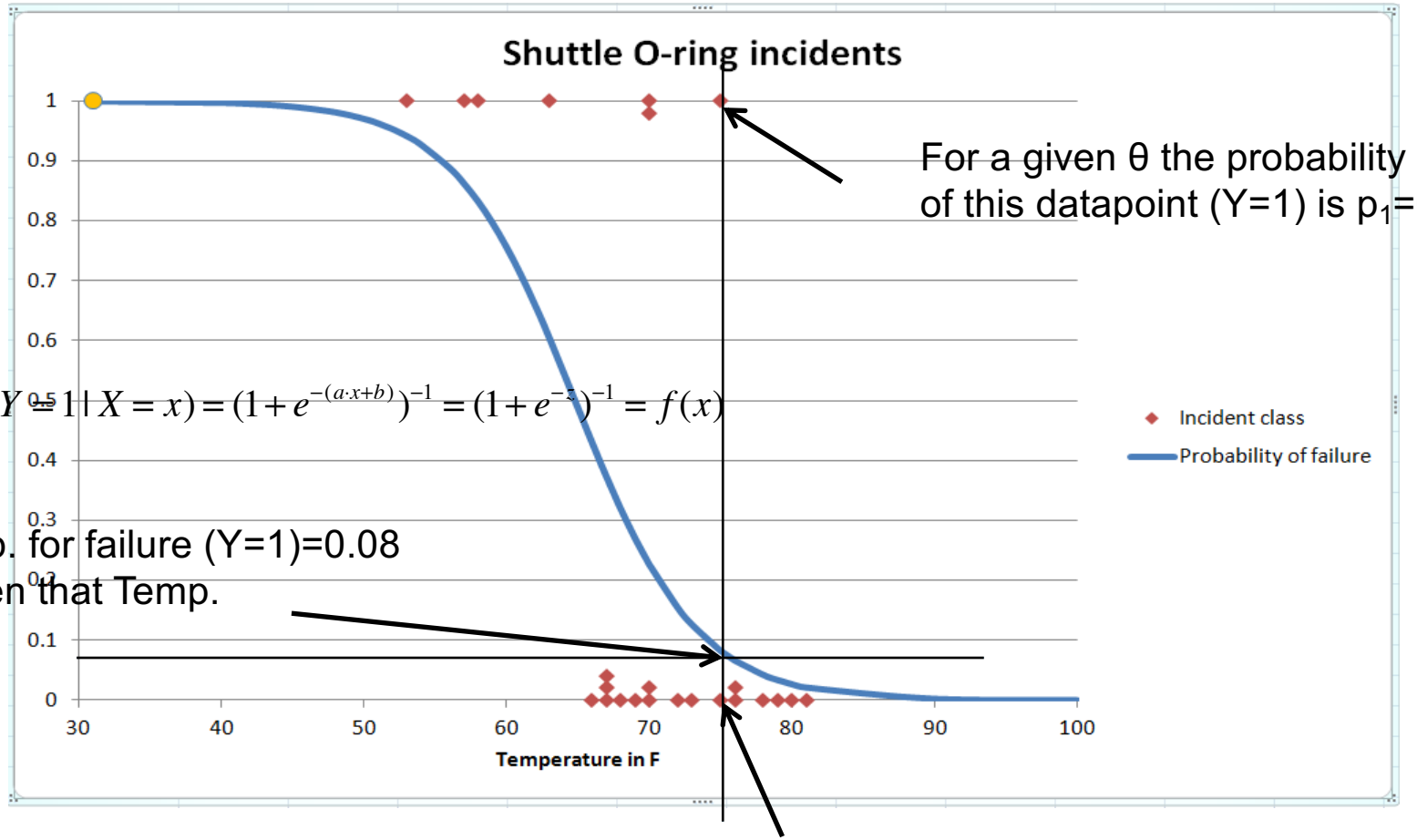


Tune the parameter(s) θ of the model M so that (observed) data is most likely

What‘s the likelihood of the data for log. regression...

Likelihood: Probability of a single observation

Two data points Y=1 (failure) and Y=0 (OK)



Y

$$p_1 = P(Y=1 \mid X=x) = (1+e^{-(a\cdot x+b)})^{-1} = (1+e^{-z})^{-1} = f(x)$$

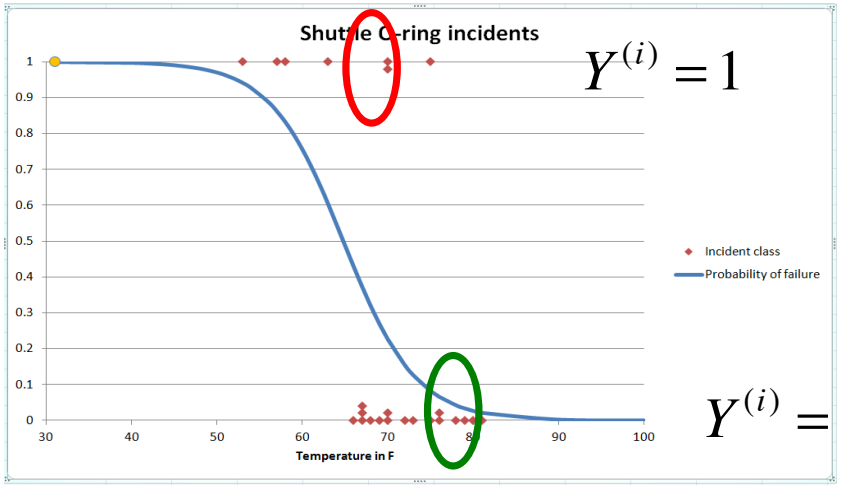
Prob. for failure (Y=1)=0.08
Given that Temp.

For a given θ the probability of this datapoint (Y=1) is p_1=0.08

For a given θ the probability of this datapoint (Y=0) is 1 - 0.08 = 92%

Prob. of all data points is the product of the individual data points...
(if iid).

Likelihood: Probability of the training set



$Y^{(i)} = 1$

$Y^{(i)} = 0$

Training Data i = 1...N

$X^{(i)}, Y^{(i)}$

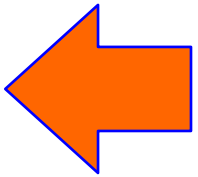
$$p_1(X) = P(Y = 1 \mid X) = (1 + e^{-(a \cdot x + b)})^{-1} = (1 + e^{-z})^{-1} = f(x)$$

Probability to find Y=1 for a given values X (single data point) and a, b

$p_0(X) = 1 - p_1(X)$ Probability to find Y=0 for a given value X (single data point)

Likelihood (probability+ of the training set given the parameters)

$$L(a,b) = \prod_{i \in All\ ones} p_1(x^{(i)}) * \prod_{i \in All\ Zeros} p_0(x^{(j)})$$



Let‘s maximize this probability

Maximizing the Likelihood

Likelihood (prob of a given training set) want to maximized wrt. parameters

$$L(a,b) = \prod_{i \in All\ ones} p_1(x^{(i)}) \quad * \quad \prod_{i \in All\ Zeros} p_0(x^{(j)})$$

Taking log (maximum of log is at same position)

$$-NJ(\theta) = L(\theta) = L(a,b) = \sum_{i \in All\ ones} \log(p_1(x^{(i)})) \quad + \quad \sum_{i \in All\ zeros} \log(p_0(x^{(i)})) = \sum_{i \in All\ Training} y_i \log(p_1(x^{(i)})) + (1 - y_i)\log(p_0(x^{(i)}))$$

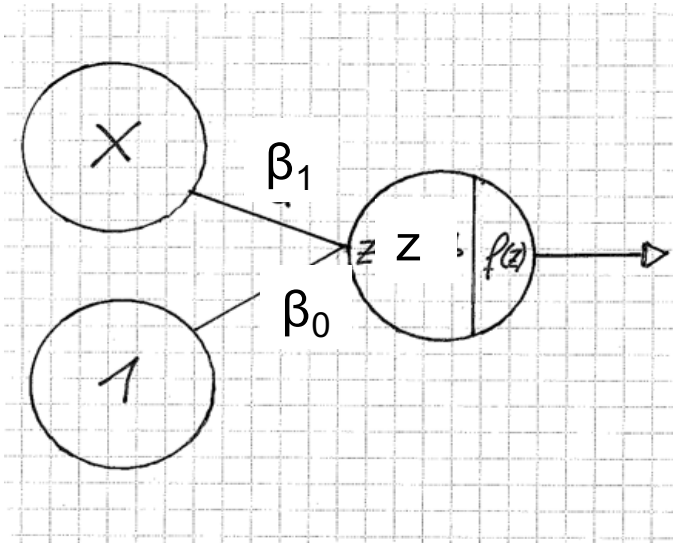
Loss function (negative log-likelihood, cross-entropy)

$$\text{loss} = -\frac{1}{N}\sum_{i=1}^{N} \log(p_{\text{model}}(y^{(i)} \mid x^{(i)}; \theta))$$

This is the prob. the model evaluates for the true class $y^{(i)}$ of training example $x^{(i)}$

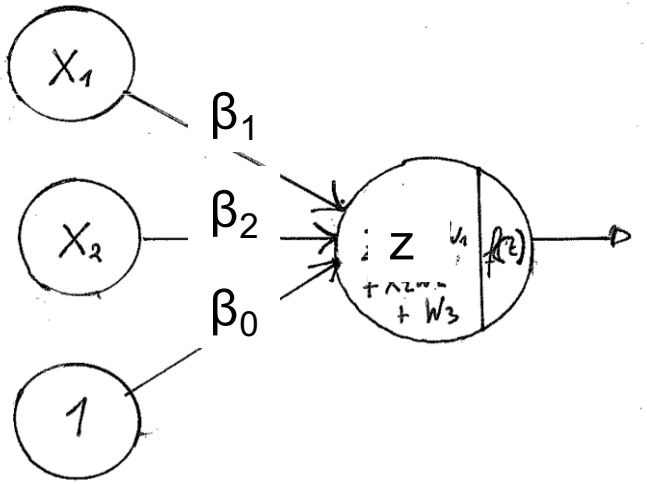
Logistic Regression the mother of all neural networks

1-D log Regression



$$z = \beta_0 + \beta_1 x$$

Multivariate Log.-Regression



$$z = \beta_0 + x_1\beta_1 + x_2\beta_2 = \beta^T x$$

$$p_1(x) = P(Y = 1 \mid X = x) = [1 + \exp(-\beta^T x)]^{-1} = \frac{\exp(\beta^T x)}{1 + \exp(\beta^T x)} = f(\beta^T x)$$

Interpretation with odds ration

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}.$$

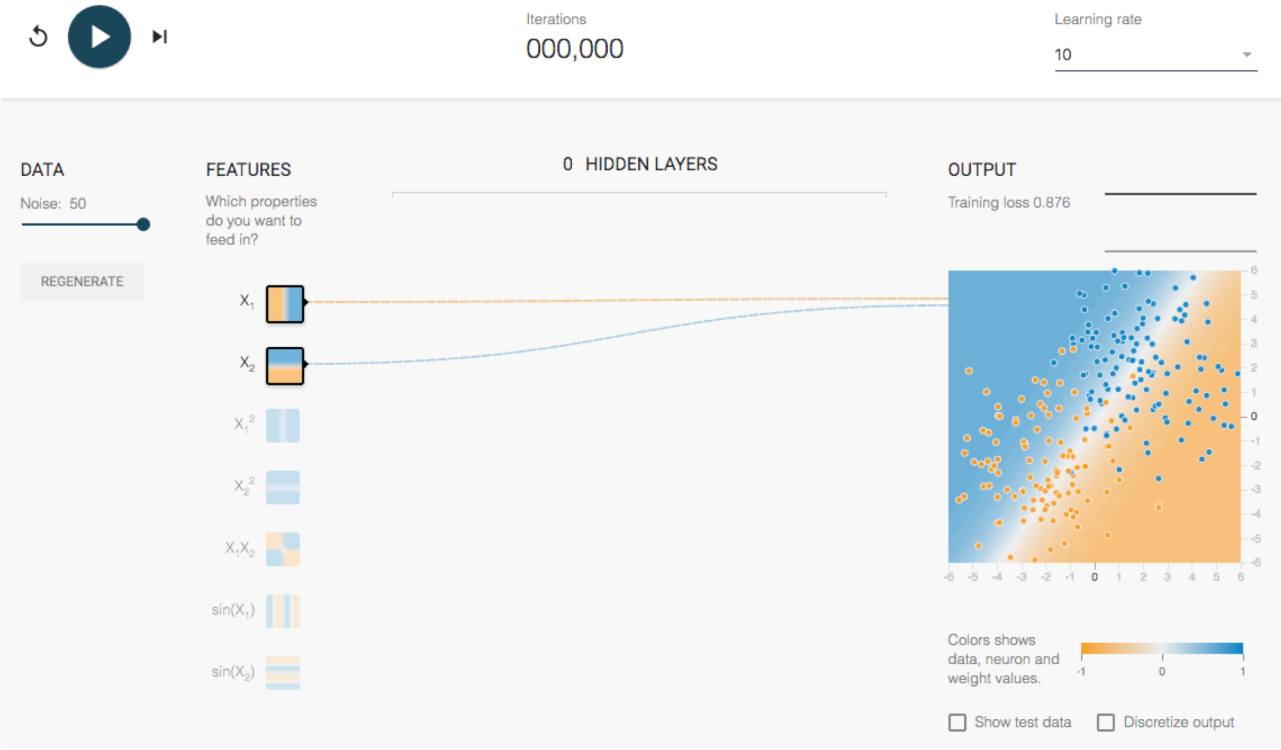
$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

- Odds : „Prob for winning“ / „Prob for not winning“ (think of horse races)
- Modeling the log odds as linear regression
- Interpretation of coefficients
- Linear Boundary

Logistic Regression [Demo]

Open the [tensorflow playground](#) and

a) manually adjust the the weights to minimize the training loss?
b) Start learning with a learning rate 10 what happens?
c) Change learning rate to sensible values

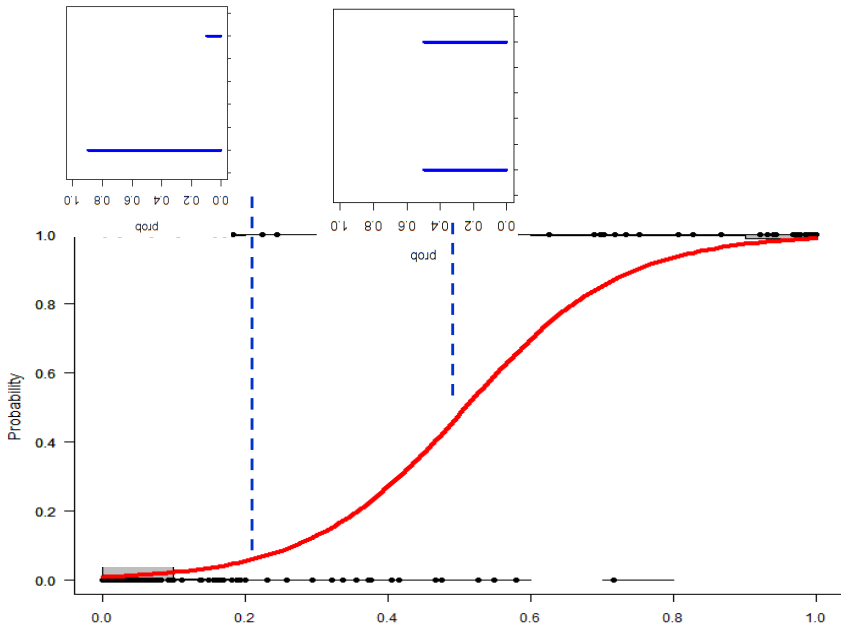


Comparison Logistic Regression / Linear

Logistic Regression

$$param = a \cdot x + b$$
$$Y \sim Bern(p = \text{sig}(para))$$

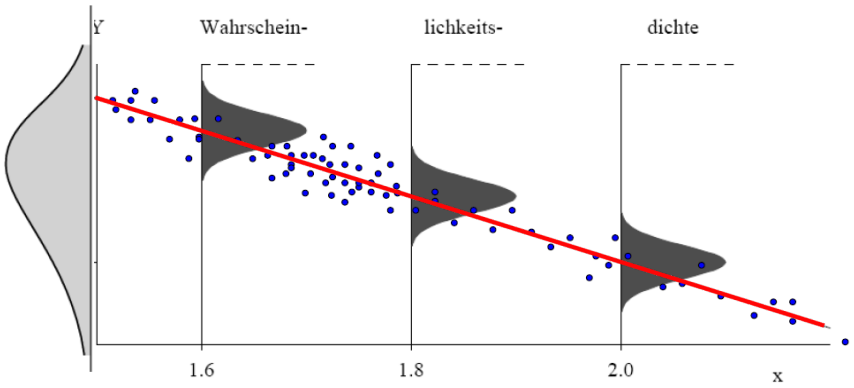
```
glm(y ~ ., binomial(logit))
```



Linear Regression

$$param = a \cdot x + b$$
$$Y \sim Normal(\mu = para, \sigma = 1)$$

```
glm(y ~ ., gaussian(identity))
```



Further prob. models

Poisson Regression

$$param = a \cdot x + b$$
$$Y \sim Poiss(rate = \mathrm{Exp}(para))$$

```
glm(y ~ ., poisson(log))
```

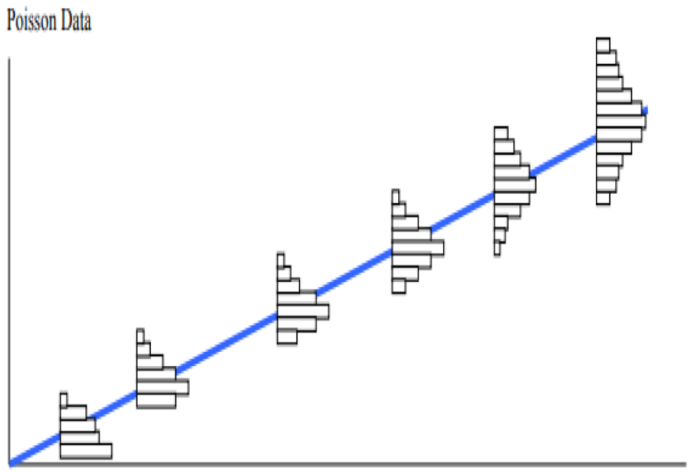


Image credit: https://www.lexjansen.com/nesug/nesug10/sa/sa04.pdf

General Framework

1. $para = a \cdot x + b$ #ext. to more
2. Some manipulation of param
3. Control prob. Distribution

- Generalized linear models
- Neural networks
- Can all be solved with gradient descent*

* (there are specialized methods for GLMs, IRLS)